Model-Based Design for High Integrity Software Development

Mike Anthony
Senior Application Engineer
The MathWorks, Inc.
Tucson, AZ USA
Model-Based Design for High Integrity Software Development

Agenda

Development and V&V of the Model

- Building a Model from Requirements
  - Introduction to Simulink

- Traceability of a Model to Requirements
  - Using the Requirements Management Interface
  - The Requirements Report

- Conformance to Modeling Standards
  - Using the Model Advisor
  - Customizing the Model Advisor
  - Model Advisor Report

- Verification of the Model against Requirements
  - Introduction to SystemTest
  - SystemTest Report
  - Introduction to Simulink Design Verifier: Property Proving

Development and V&V of the Code

- Production Code Generation
  - Creating Data Objects
  - Function Prototype Control

- Traceability of the Generated Code to the Model
  - Code-to-Model Linking
  - Model-to-Code Linking
  - Traceability Report

- Conformance to Coding Standards & Code Verification
  - PolySpace
    - MISRA-C Compliance
    - Proving the Absence of Runtime Errors

- Verification of the Generated Code against the Model
  - Introduction to Simulink Design Verifier: Test-Vector Generation
    - SystemTest
    - Embedded IDE Link Products for PIL

- Verification of the Generated Code against the Requirements
  - SystemTest: Test Case reuse
  - Embedded IDE Link Products for PIL
Aerospace Standards

- **RTCA/DO-178B Guidelines**
  - Commercial standard (FAA, JAA)
  - Software Integrity Levels A-E based on hazards
  - Level A if failure hazards can cause loss of life or limb
  - Structural coverage (MC/DC)

- **UK MOD 0055/0056**
  - Software Integrity Levels 1-4
  - Requires formal analysis and software proofs
  - Has SPARK language and data flow checks

- **MIL-STD-498**
  - Formerly DOD-2167A
  - US military and defense
  - Emphasizes verification and validation activities
Methods for Verification and Validation

Verification: Did I do the design right?
Validation: Did I do the right design?

- **Traceability**
  - Requirements to model and code
  - Model to code

- **Modeling and Coding Standards**
  - Modeling standards checking
  - Coding standards checking

- **Testing**
  - Model testing in simulation
  - Processor In the loop

- **Proving**
  - Proving design properties
  - Proving code correctness
Workflow Example

- **Requirements**
  - Validate
- **Model**
  - Conformance
  - Verify
- **Source Code**
  - Trace
  - Conformance
  - Verify
- **Object Code**
  - Verify
Workflow Example

The MathWorks

Verify
Simulink Design Verifier:
Property Proving
Model Coverage

SystemTest*
Embedded IDE Link

Verify
SystemTest*
Test Generation
Embedded IDE Link

Verify
SystemTest*

Model Advisor*

Conformance
PolySpace*

Simulink & Stateflow

Simulink Verification and Validation: Requirements Management Interface

Trace

Source Code

Object Code

Source Code

Model

Requirements

Real-Time Workshop Embedded Coder

Model/Code Trace Report

Trace

* DO-178B Qualifiable Tool
Workflow Example

Trace
Simulink Verification and Validation: Requirements Management Interface

Validate

Requirements

Simulink & Stateflow

Conformance
Model Advisor*

Simulink Design Verifier: Property Proving
Model Coverage

Verify
SystemTest*
Embedded IDE Link

Verify
SystemTest*

Simulink Design Verifier: Test Generation
Embedded IDE Link

Verify

Model

Conformance
PolySpace*

PolySpace*

Verify
SystemTest*

Simulink Design Verifier: Property Proving
Model Coverage

Verify
SystemTest*

Simulink Design Verifier: Test Generation
Embedded IDE Link

* DO-178B Qualifiable Tool

Trace
Model/Code Trace Report

Real-Time Workshop Embedded Coder

Trace

Source Code

Embedded IDE Link

Object Code

* DO-178B Qualifiable Tool
Requirements Management Interface Overview

- Associating models and requirements
  - Establishing a link from a model block or test case to requirement
  - Establishing a link from a requirement to a model block or test case
- Managing changes in models and requirements
  - Detecting a change in a requirement associated with a model block or test
  - Detecting a change in a model block or test associated with a requirement
- Reporting requirement coverage for model blocks and test cases
  - How many and what algorithmic blocks are covered by requirements
  - How many and what test cases are covered by requirements
  - How many of the requirements are covered by test cases within the model
  - How many of the requirements are associated with algorithmic blocks within the model
Workflow Example

Trace
Simulink Verification and Validation: Requirements Management Interface

Validate
Requirements

Simulink & Stateflow

Conformance
Model Advisor*

Simulink Design Verifier: Property Proving Model Coverage
SystemTest*

Conformance
PolySpace*
PolySpace*

Model

Verify
SystemTest*
Embedded IDE Link

Real-Time Workshop Embedded Coder

Source Code

Verify
SystemTest*
Simulink Design Verifier: Test Generation Embedded IDE Link

Verify
SystemTest*

Object Code

Trace
Model/Code Trace Report

Embedded IDE Link

* DO-178B Qualifiable Tool
Simulink Model Advisor

- Model Advisor is used to
  - Enforce model standards and best practices
  - Detect and troubleshoot modeling and code generation issues
  - Check models for (a subset of) known version upgrade issues
Model Advisor Within Model-Based Design Workflows

- Requirements
  - Missing requirements
  - Invalid requirement links

- System Architecture
  - Simulation accuracy
  - Modeling style
  - Modeling errors
  - Safety issues
  - Design efficiency
  - Feature misuse
  - Invalid library links

- Prototype Design Models
  - Incorrect code generation
  - Inefficient code generation
  - Implementation issues

- Refined Models

- Code Generation

- Component Source Code

- Target Application

- System Integration

- MathWorks Automotive Advisory Board Checks

- Right click to select/deselect all

- Analysis
  - Run Selected Checks
  - Show Report After Run

- Last Report
  - From node: MathWorks Automotive Advisory Board Checks
  - Report: C:\Work\Art\start\model Advisor\txt\adv
  - Date/Time: 04-Jun-2007 21:58:21
  - Summary: Pass: 10 Fail: 0
  - Not Relevant: 0
Simulink Verification and Validation
Additional Model Standards Checking

- DO-178B Checks
  - Focus on generation of safety critical code from models
  - Assist in MISRA-C compliance
  - Maximize traceability of code to model
  - Minimize differences between model coverage and code coverage
  - Maximize the use of built-in Simulink and Stateflow diagnostics during simulation

- MathWorks Automotive Advisory Board Checks
  - Simulink style guide created by MathWorks Automotive Advisory Board
  - Best practices for consistent and readable models
Model Advisor Report

Report enhanced to be more useful as a process audit
Document:

- More detailed summary
- Report follows exact order of the Model Advisor tree
- Valid check states: Pass, Fail, Warning, and Not Run
Model Advisor
Enterprise Deployment

- The Model Advisor is highly customizable:
  - Add additional task groups and checks
  - Permanently enable disable, and hide specific checks

- Benefits
  - Enforce your specific process and standards
  - Prevent defects at specific points early in your design process
Workflow Example

**Trace**
- Simulink Verification and Validation: Requirements Management Interface

- Model/Code Trace Report

- Real-Time Workshop Embedded Coder

**Validate**
- Requirements

- Model

- Source Code

- Object Code

- Simulink & Stateflow

**Conformance**
- Model Advisor*

- PolySpace*

**Verify**
- SystemTest*

Simulink Design Verifier: Property Proving

Model Coverage

- SystemTest*

Simulink Design Verifier: Test Generation

Embedded IDE Link

* DO-178B Qualifiable Tool
SystemTest Software

- Manage tests and analyze results for system verification and validation

**Simulink System Model**

1. **Load**
2. **Setup Test and Variables**
3. **Run Simulations**
4. **Analyze Results**
SystemTest

Key Features

- Develops, manages, and edits test structures using predefined test elements in a graphical user interface
- Stores tests in a separate TEST-file independent of the model under test for repeatable test execution
- Defines pass/fail criteria for tests using Boolean constraints and tolerance limits
- Generates random test vector values using probability distribution functions, especially useful for Monte Carlo simulations
- Runs iterations, such as parameter sweeps, of Simulink models on multiple processors with Distributed Computing Toolbox (available separately)
- Generates reports of test execution and results
- Visualizes and analyzes multidimensional test results in Test Results Viewer
SystemTest
Sample Applications

- Stress testing
- Parameter sweeps
- Model verification and validation
  - Vary block parameters
  - Measure and report model coverage (with Simulink Verification and Validation)
- Algorithm verification and validation
- Monte Carlo simulation
Workflow Example

Trace
Simulink Verification and Validation: Requirements Management Interface

Validate
Requirements

Simulink & Stateflow
Model

Conformance
PolySpace*
Model Advisor*

Conformance
Model Coverage

Verify
SystemTest*
Simulink Design Verifier:
Property Proving

Test Generation
Embedded IDE Link

Source Code

Embeded IDE Link

Object Code

Trace
Model/Code Trace Report

Real-Time Workshop
Embedded Coder

* DO-178B Qualifiable Tool
Simulink Design Verifier
Property Proving

- Functional testing
  - Generates a proof for a requirement
    - For example: Thrust reversers shall not deploy in flight
  - Includes blocks for definition of properties
  - Proves model properties and generates example of violations
  - Produces detailed property-proving analysis reports

- Uses formal methods, not simulation
Property Proving
Verification Results

- Proof or assertion can be found:
  - Satisfied
  - Falsified
  - Undecidable

- If Falsified, a test case is generated and added to the model harness
Model Coverage
Measure of Test Completeness

- Execution analysis
  - Based on the model structure
  - Dynamic – data collected during simulation

- Coverage results
  - Displayed directly in the model
  - Available in a separate HTML report linked with the model objects

- Supports
  - Simulink
  - Stateflow
  - Embedded MATLAB

Supported coverage types
- Decision coverage
- Condition coverage
- MC/DC
- Lookup table coverage
- Signal range coverage
Workflow Example

**Trace**
- Simulink® Verification and Validation™: Requirements Management Interface
- Model/Code Trace Report

**Validate**
- Requirements
- Model
- Source Code
- Object Code
- Simulink® & Stateflow®
- Real-Time Workshop® Embedded Coder™
- Embedded IDE Link™

**Conformance**
- Model Advisor*
- PolySpace®

**Verify**
- SystemTest*
- PolySpace®* (DO-178B Qualifiable Tool)
- Simulink Design Verifier: Property Proving
- Model Coverage
- Test Generation
- Embedded IDE Link

*DO-178B Qualifiable Tool
Model-to-Code and Code-to-Model Traceability

- Use Simulink Verification and Validation software to navigate and trace between model elements and requirements.

- Use Real-Time Workshop Embedded Coder software to navigate and trace between generated code back and its source model.

```c
/* Exported block signals */
real_T INPUT;
real_T OUTPUT;

/* Exported block parameters */
real_T k = 5.0;

OUTPUT = INPUT * k;
```
Traceability Report
Real-Time Workshop Embedded Coder

- Use the Traceability Report section of the Real-Time Workshop Embedded Coder code generation report to review mapping between model elements and generated code.
Workflow Example

Trace
Simulink Verification and Validation: Requirements Management Interface

Object Code

Source Code

Model

Requirements

Simulink & Stateflow

Conformance
Model Advisor*

Verify
SystemTest*

Embedded IDE Link

SystemTest*

Simulink Design Verifier:
Property Proving
Model Coverage

Simulink Design Verifier:
Test Generation
Embedded IDE Link

Verify
SystemTest*

Conformance
PolySpace*

* DO-178B Qualifiable Tool

Model/Code Trace Report

Real-Time Workshop Embedded Coder

Embedded IDE Link
Why prove the absence of run-time errors?
Implications of verification, static analysis & unit testing

Number of operations \times input values

0% proven reliable

Verification

Static analysis

Testing

T0

+3 months

+6 months

Improvement is possible and measurable

One-time improvement, but nothing measurably proven

Required for functional testing. Not suitable to prove code correctness
Code Correctness

Formal method: Abstract Interpretation

Results are proven for all possible executions of the code!!
PolySpace Link

Solution

- Trace run-time errors back to the model
- Integrate code verification into the production code generation
Workflow Example

Trace
Simulink Verification and Validation: Requirements Management Interface

Validate

Requirements

Simulink & Stateflow

Model

Conformance
Model Advisor*

Conformance
PolySpace*

Real-Time Workshop
Embedded Coder

Source Code

Object Code

Simulink Design Verifier: Property Proving
Model Coverage

Simulink Design Verifier: Test Generation
Embedded IDE Link

Verify
SystemTest*

Verify
SystemTest*

Verify
SystemTest*

* DO-178B Qualifiable Tool
Simulink Design Verifier

Test Generation

- Automatically generates test vectors to achieve 100% coverage
- Detects unreachable states
- Save test vectors
  - Automatically generate a separate model with test harness
  - Export test vectors to .CSV file
- Automatically generates test vector report
  - Two-way mapping of objectives and generated vectors
    - List of objectives and associated test vector
    - List of test vectors and associated objective(s)
Simulink Design Verifier

Test Generation

- Verify model satisfies requirements
  - Find test vectors for coverage not achieved by functional tests
    - Create unspecified requirement
    - Remove model function not traceable to a requirement

- Verify object code functions according to model
  - Generate test vectors for model coverage
  - Execute test vectors on model
  - Execute test vectors on object code
  - Compare model and code outputs for equality
Workflow Example

Trace
Simulink Verification and Validation: Requirements Management Interface

Validate
Simulink & Stateflow
Conformance
Model Advisor
PolySpace*

Verify
SystemTest*
Simulink Design Verifier: Property Proving Model Coverage
PolySpace*
Simulink Design Verifier: Test Generation Embedded IDE Link

Source Code
Conformance
Real-Time Workshop Embedded Coder

Object Code
Model/Code Trace Report

Embedded IDE Link

* DO-178B Qualifiable Tool
Processor-in-the-Loop Testing
Embedded IDE Link MU

- Model in simulation and code on the processor running in parallel

PIL also provides execution profiling, code coverage reports, and interactive debugging.
Introduction to DO Qualification Kit

- Provides documentation, test cases, and procedures that help you use Simulink or PolySpace software verification tools for projects based on the DO-178 standard
- Includes tool qualification plans, tool operational requirements, and other materials required for qualifying software verification tools
- Helps streamline certification of your embedded systems developed using Simulink or PolySpace products
Key Features

- Tool Qualification Plan and Tool Operational Requirements
- Test case models and code, test procedures, and expected results
- Traceability tables mapping test cases to requirements
- Qualification materials for Simulink verification, validation, and test tools
- Qualification materials for PolySpace code verification tools
Working with DO Qualification Kit

To use DO Qualification Kit:

1. Propose tool qualification to certification authorities.
2. Document tool operational requirements.
3. Check if the tool satisfies operational requirements:
   - Yes: Provide certification authorities with tool qualification results.
   - No: Verify that the correct version of all required software is installed correctly and is being used. Check for known bug reports related to this product, which exist on www.mathworks.com. Contact MathWorks Support for assistance. Document a limitation of tool usage in your Tool Accomplishments Summary.
Tool Qualification Plan and Operational Requirements

DO Qualification Kit 1.0 contains qualification artifacts for the following products:

- Simulink Verification and Validation
  - DO-178B Model Checks
- SystemTest
  - Limit Test Element
- PolySpace verification products
  - PolySpace Client for C/C++
  - PolySpace Server for C/C++

Version support includes:

- Release R2008b
- Release R2009a
Summary

DO Qualification Kit:

- Eases your embedded system certification process
- Helps satisfy the objectives of verification tool qualification described in DO-178B (Section 12.2)
- Facilitates automated software verification for DO-178
- Enables use of state-of-the-art development tools for Model-Based Design with flight code generation
- Enables qualification of PolySpace products, including formal analysis capabilities
Workflow Summary

Trace
Simulink Verification and Validation: Requirements Management Interface

Validate

Requirements

Simulink & Stateflow

Conformance
Model Advisor*

Simulink Design Verifier: Property Proving
Model Coverage

Simulink Design Verifier: Test Generation

Verify
SystemTest*
Embedded IDE Link

Verify
SystemTest*

Source Code

Conformance
PolySpace*

Model Advisor*

Verify
SystemTest*

Embedded IDE Link

Object Code

Real-Time Workshop Embedded Coder

Trace
Model/Code Trace Report

* DO-178B Qualifiable Tool